Scikit-learn Unsupervised Methods

> by: Saeed Mohagheghi + 🤖 AI

📉 Dimensionality Reduction Methods in scikit-learn

Method	Import Statement	Pros	Cons
Principal Component Analysis (PCA)	`from sklearn.decomposition import PCA`	Fast, widely used, captures maximum variance	Assumes linearity, components may be hard to interpret
Kernel PCA	`from sklearn.decomposition import KernelPCA`	Captures non-linear structures	Requires kernel tuning, slower than PCA
Truncated SVD (LSA)	`from sklearn.decomposition import TruncatedSVD`	Works with sparse data, good for text (LSA)	Less accurate than PCA for dense data
Independent Component Analysis (ICA)	`from sklearn.decomposition import FastICA`	Finds statistically independent components	Sensitive to noise, not guaranteed to reduce dimensionality
t-SNE	`from sklearn.manifold import TSNE`	Excellent for visualization, captures non-linear relationships	Computationally expensive, not suitable for large datasets
Isomap	`from sklearn.manifold import Isomap`	Preserves global geometry, good for non-linear manifolds	Sensitive to noise and parameter tuning
Locally Linear Embedding (LLE)	`from sklearn.manifold import LocallyLinearEmbedding`	Preserves local structure, good for manifold learning	Sensitive to noise, poor scalability
UMAP *(via third-party)*	`import umap` *(requires* `umap-learn`*)*	Fast, preserves both local and global structure	Not in scikit-learn core, sensitive to parameters

Method	Import Statement	Pros	Cons
Linear Discriminant Analysis (LDA)	`from sklearn.discriminant_analysis import LinearDiscriminantAnalysis`	Supervised, good for class separation	Requires labeled data, assumes normal distribution
Feature Agglomeration	`from sklearn.cluster import FeatureAgglomeration`	Hierarchical clustering of features, interpretable	Less commonly used, may lose fine-grained structure

---

🧠 **Tips**:

- Use `.fit_transform(X)` to reduce dimensions.
- PCA and Truncated SVD are great for preprocessing before modeling.
- t-SNE and UMAP are ideal for visualizing high-dimensional data in 2D or 3D.
- LDA is supervised and best used when class labels are available.

🔍 Clustering Methods in scikit-learn

Clustering Algorithm	Import Statement	Pros	Cons
K-Means	`from sklearn.cluster import KMeans`	Fast, scalable, easy to implement	Assumes spherical clusters, sensitive to initial centroids
DBSCAN	`from sklearn.cluster import DBSCAN`	Detects arbitrary-shaped clusters, handles noise	Struggles with varying densities, sensitive to parameters
Agglomerative Clustering	`from sklearn.cluster import AgglomerativeClustering`	No need to specify number of clusters, interpretable dendrograms	Computationally expensive for large datasets
Mean Shift	`from sklearn.cluster import MeanShift`	Automatically finds number of clusters, handles non-linear shapes	Slow, memory-intensive
Spectral Clustering	`from sklearn.cluster import SpectralClustering`	Good for complex cluster structures, graph-based	Not scalable to large datasets, requires affinity matrix

Clustering Algorithm	Import Statement	Pros	Cons
Affinity Propagation	`from sklearn.cluster import AffinityPropagation`	No need to predefine number of clusters	Slow, high memory usage, sensitive to preference parameter
Birch	`from sklearn.cluster import Birch`	Scales well to large datasets, incremental learning	Assumes convex clusters, less effective on non-spherical data
OPTICS	`from sklearn.cluster import OPTICS`	Handles varying densities, robust to noise	Slower than DBSCAN, complex parameter tuning
Gaussian Mixture (GMM)	`from sklearn.mixture import GaussianMixture`	Probabilistic clustering, flexible cluster shapes	Assumes Gaussian distribution, sensitive to initialization

---

🧠 **Tips**:

- Clustering is unsupervised: no labels (`y`) are used.
- Use `.fit(X)` or `.fit_predict(X)` to apply clustering.
- Visualization (e.g., with PCA or t-SNE) often helps interpret clusters.
- For high-dimensional data, consider dimensionality reduction before clustering.